# GROUP 14 ASSIGNMENT 2

CSY2038 PR1 BUILD

DATABASES 2

Submission Date: /01/2022

Authors:

Carlos Bristol (20402942)

Emily Fletcher (18410839)

Maximus Kelly (19424942)

Rye Kettless (20412467)

Peer Review:

EQUAL MARKS

Or

Carlos Bristol: %

Emily Fletcher: %

Maximus Kelly: %

Rye Kettless: %

Agreement Signatures:

# Contents

Section One

* Introduction and Schema
* Skeleton Tables
* Naming Conventions
* Data Extraction Methods
* Additional Research and Automation
* Purpose of Procedures, Functions and Triggers (Author Note Must have comments)

Section Two

* Test Plans
* Test Cases
* Testing and Outcomes

References

Appendix

* Code Appendices (Presented as Text)

Section One

Introduction and Schema

The assignment brief outlines the main learning outcomes of this assignment to be a database project that demonstrates an understanding of data definition and data manipulation using procedural SQL. This document is part of the deliverables requested by the assignment setter and covers requirements task one as well as task three. With the appendix covering task two. The overall aim of this deliverable is to broadcast group fourteens understanding of the design, research, implementation, and testing stages of creating a complex multi-table relational database.

After examining the Entity Relationship Diagram sent with the brief group fourteen decided on a subsection to explore to build a relational database around. This specific section was selected due to their close connectivity which was believed to have a great potential for expansion and complex relation demonstration. The ERD was expanded upon by adding objects, Entities within the diagram are shown as boxes with rounded edges, whereas objects are shown with standard boxes. Entity relations have been represented by crows’ feet and data flow has been shown by arrows between objects. The section of ERD is shown in figure one.

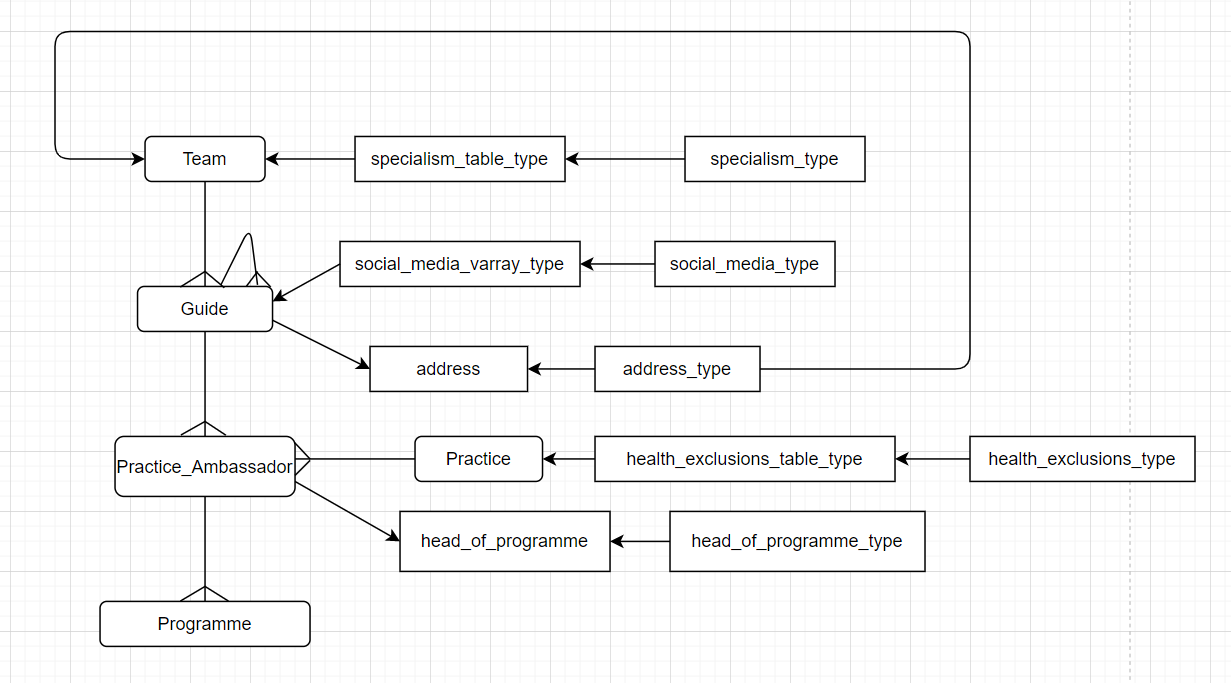


Figure Group 14's Expanded ERD

Skeleton Tables

Five relational tables were planned into the skeleton model as well as five objects. The first skeleton tables represent entities contained within is the name of the table, which is plural to keep in guidance with best practice, the attributes that will be within that table, any allocation of keys, such as the primary key and foreign key, the data type of that attribute and any constraints that are tied to that attribute. The second skeleton table represents all the objects from the ERD. This is broken down into the Object itself being a Unique Data Type, an Object Table, or a Varray. As well as its type, it’s attributes, and the data types of the attributes.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Table** | **Attributes** | **Key** | **Datatype** | **Constraints** |
| teams | team\_id | PK | NUMBER(6) |  |
|  | address |  | REF OF addresses | address\_type |
|  | team\_location |  | VARCHAR2(50) |  |
|  | number\_of\_team\_members |  | NUMBER(3) |  |
|  | specialism |  | specialism\_table | specialism\_table\_type |
|  | | | | |
| guides | guide\_id | PK | NUMBER(6) |  |
|  | team\_id | FK | NUMBER(6) |  |
|  | supervisor\_id |  | NUMBER(6) |  |
|  | guide\_firstname |  | VARCHAR2(35) |  |
|  | guide\_surname |  | VARCHAR2(35) |  |
|  | guide\_date\_of\_birth |  | DATE |  |
|  | date\_of\_employment\_start |  | DATE |  |
|  | time\_employed |  | NUMBER(5) | (insert function here) |
|  | address |  | REF OF addresses | address\_type |
|  | phone\_number |  | VARCHAR2(11) | UNIQUE |
|  | social\_media |  | social\_media\_varray  \_type |  |
|  | | | | |
| practices | practice\_id | PK | NUMBER(6) |  |
|  | practice\_title |  | VARCHAR2(50) |  |
|  | practice\_description |  | VARCHAR2(50) |  |
|  | duration\_hours |  | NUMBER(3) | DEFAULT 1 |
|  | practice\_location |  | VARCHAR2(50) |  |
|  | health\_exclusions |  | health\_exclusions\_table | health\_exclusions\_table\_type |
|  | | | | |
| practice\_ambassadors | ambassador\_id | PK | NUMBER(6) |  |
|  | practice\_id | FK | NUMBER(6) |  |
|  | guide\_id | FK | NUMBER(6) |  |
|  | head\_of\_programme |  | head\_of\_programm\_type |  |
|  | | | | |
| programmes | programme\_id | PK | NUMBER(6) |  |
|  | ambassador\_id | FK | NUMBER(6) |  |
|  | programme\_title |  | VARCHAR2(50) |  |
|  | programme\_total\_cost |  | NUMBER(10,2) |  |
|  | programme\_start\_date |  | DATE |  |
|  | programme\_end\_date |  | DATE |  |
|  | programme\_duration\_days |  | NUMBER(2) | Insert Function here |
|  |  |  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **Object** | **Object Type** | **Attributes** | **Data Types** |
| addresses | address\_type | Street | VARCHAR2(50) |
| City | VARCHAR2(35) |
| Country | VARCHAR2(35) |
|  | | | |
| specialism\_table\_type | specialism\_type | specialism\_title | VARCHAR2(35) |
| specialism\_type | VARCHAR2(35) |
| specialism\_description | VARCHAR2(100) |
|  | | | |
| health\_exclusions\_  table\_type | health\_exclusions\_type | exclusion\_title | VARCHAR2(35) |
|  | exclusion\_description | VARCHAR2(100) |
| exclusion\_rationale | VARCHAR2(200) |
|  | | | |
| social\_media\_  varray\_type | social\_media\_type | social\_media\_platform | VARCHAR2(35) |
| social\_media\_handle | VARCHAR2(50) |
|  | | | |
| head\_of\_programmes | head\_of\_programme\_type | head\_of\_programme\_firstname | VARCHAR2(35) |
| head\_of\_programme\_surname | VARCHAR2(35) |

Naming Conventions

Naming Conventions refer to a set of rules that are decided before database creation, these are rules that focus on consistency within the database. As databases become increasingly complex it is important to maintain consistency and have a plan on how elements of the database are identified. This makes identification of components easy; this can help when fetching data, reading data and throughout data debugging. Emil Drkusic (2020). As requested, Oracle naming conventions have been applied as there are many SQL standard naming conventions.

Research for table conventions was obtained from the following resources, <https://www.red-gate.com/simple-talk/blogs/sql-naming-conventions/>

<https://oracle-base.com/articles/misc/naming-conventions>

<https://docs.oracle.com/cd/E18727_01/doc.121/e12897/T302934T458266.htm>

<https://www.sqlshack.com/learn-sql-naming-conventions/>

|  |  |
| --- | --- |
| **Item Discussed** | **Convention Applied** |
| General Rules | Underscores are used to make spaces, do not use spaces, avoid numbers where possible for attributes and names, do not use reserved words when naming, avoid use of special. characters or symbols when creating the database. For naming user lowercasing, for data use UPPERCASING. Avoid alternate casing and camelCasing. Commands should also use UPPERCASING e.g. CREATE TABLE teams.  Where possible headings and wraps should be used when data overflows to ensure tables are clear and easy to read, however this is not essential. |
| Comments | The main file should have a header that contains the file name, authors, student number, group number, module date and version number contained without multiple line comments.  The run command should be underneath the header comments within a single line comment.  Each new section of the file should have a titular comment that uses the same conventions as the content below it, these should be single line comments such as  --CREATE TABLE COMMANDS  -- teams TABLE,  These help ease navigation.  Comments should not be applied mid-way through database code as this can break the database e.g CREATE TABLE teams ( -- Teams TABLE.  Comments used to explain concepts should be underneath the titular comment in multi-line comments |
| Entities | All entities are plural nouns for names as they store multiple records inside of them, e.g Customers, Programmes. Avoid long names where possible, ideally one or two words. In this case of more than one word only the last word of the table name should be plural Avoid numbers when naming tables, e.g customers1. |
| Attributes | Attributes should be named singularly to avoid confusion with entity names. Where possible attributes should have unique names to make joining tables easier. Where attributes are repeated across tables (or likely to be repeated if the database is expanded) the tables name in its singular form should be a prefix for the attribute eg. guide\_firstname.  Attributes should not use shorthand naming as this can be confusing and unclear on what a attribute is. e.g no\_of\_members vs number\_of\_members. |
| Primary Keys | Primary keys should have the prefix pk\_ and be named after the table they are in. E.g pk\_teams |
| Foreign Keys | Foreign keys should have the prefix fk\_ followed by the alias for the table it resides in. Its name should be the name of the table it references E.g fk\_g\_teams which resides in the guides table and references the team table. |
| Aliases | Where possible Aliases should be single characters that share the same first letter as the entity. Eg t is the alias for the teams table. Where this causes repeated aliases the second or even third letter should be added. In cases of severe alias repetition then entity names should be reconsidered. |
| Sequences | Sequences should be names with the prefix seq\_ followed by the name of the attribute that is to have the sequence applied to it eg. seq\_team\_id |
| Object Tables | Object tables are referred to as plural nouns, the same as standard entities. |
| Types | Types have the suffix \_type to distinguish them |
| Varrays | Varrays have suffix \_varray\_type to distinguish them |
| Nested Tables | Nested tables have the suffix table\_type |
| Inserted Data | Textual data that is inserted into the table is inside single quotes and is in UPPERCASE |
| Functions |  |
| Procedures |  |
| Loops |  |
| Triggers |  |
| Cursors |  |

Data Extraction Methods

Data Extraction refers to the process of selecting specific data in the database that can be used for further analysis or future procedures.

Talend (201X) defines Data Extraction as *“the process of collecting* *or retrieving disparate types of data from a variety of source”*. This relates to this assignment as there may be specific information someone viewing the database wishes to know, however looking though the whole database is not an efficient way to do this, even with a clear database as demonstrated in our above skeletons. In this scenario the wide range of sources being the different tables and objects that are planned to be created. The benefits of Data Extraction are mentioned in this Microsoft article about queries (201X) 🡨 Find better resource if you can as this discusses the use of Access which I’m sure will piss off Carole. The article states that queries can be used to view certain records from a selection, allow the combination of different tables to find similarities or differences, as well as allowing advanced formatting and displays.

Data Extraction within this assessment is performed using Queries. A plan of queries will be demonstrated below to demonstrate a range of data extraction methods. A variety of data extraction methods is important to demonstrate a range of knowledge and data selection options.

|  |  |
| --- | --- |
| **Query** | **Query Type** |
| Find where all the teams in the retreat are located | Simple Projection Query |
| Find all the guides ID number, first name and surname | Simple Projection Query |
| Find the dates for when all programmes start and end | Simple Projection Query |
| Find all the teams who have a small team size (less than ten members) | Simple Restriction Query |
| List all of the programme heads who are not ‘Sarah Smith’ | Simple Restriction Query |
| Find all the staff named David who were born after 01/01/1999 | Multiple Restriction Query |
| Find all the practices that take place in the Sauna and are over one hour in length | Multiple Restriction Query |
| Find the addresses of all guides | Unrestricted Ref Query |
| Find the addresses of all the guides born after the 26th of November 2004 | Restricted Ref Query |
| Find all the guides with social media, what social media they use and their handles | Unrestricted VARRAY Query |
| Find all the guides who use Instagram for their social media | Restricted VARRAY Query |
| Find all the teams and their specialisms | Unrestricted Nested Query |
| Find all the teams who have the art specialism | Restricted Nested Query |
| Find all the practices and their health exclusions | Unrestricted Nested Query |
| Find all the practices with the health exclusion heart disease | Restricted Nested Query |
| Find all the guides who have a surname beginning with A | Like Query |
| Find all the teams with between 10 and 15 members | Between Query |
| Find all the practices that are one or three hours long | In query |
| Find all guides with an ID before 103 or those with a first name that starts with D | Or query |
| Find all the teams and guides that live within the UK | Union Query |
| Finds all guide\_ids that have the firstname Malcolm in guides that are in ambassadors | Intersect Query |
| Check guide\_id’s that do not have Malcolm as firstname and aren’t in ambassador | Minus Query |
| Find all firstnames in guides that are older than 18 | All Query |
| Find any firstnames in guides that have a date of birth between 26-APR-2000 and 26-APR-2001 | Any/Some Query |
| The largest team size | Maximum Function |
| The average duration of all practices | Average Function |
| The smallest team size | Minimum Function |
| The total duration of hours for a single practice | Sum Function |
| The total number of guides employed | Count Function |
| Group practice\_id and Order By DESC COUNT(practice\_id) | Group By and Order By |
| Group practice\_id HAVING COUNT(practice\_id) = 2 | Group By and Having |
| Joins practices and practice\_ambassadors with related column practice\_id | Inner Join |
| Joins teams and guides but only return values from teams and the intersection between | Left Outer Join |
| Joins practices and practice\_ambassadors but only return values from practice\_ambassadors and the intersection between | Right Outer Join |
| Joins teams and guides and returns all values from both tables | Full Outer Join |
| Returns team\_id from teams In nested query guides | Nested Query |
| Returns team\_location from teams Not In nested query guides | Negative Query |
| Using the proramme table, display programme\_id, title , total\_cost and ambassador\_id where it equals the ambassador\_id in ambassadors | Nesting for Pseudo Tables |

Additional Research and automation

Database Automation is defined as the use of unattended process and self-updating procedures for administrative tasks within a database. WhatIs.com(2019). This means that once automation is added to the database then tasks can be performed without a developer’s input. There are many benefits to this. Muhammed Raza(2021) states that a lot of operations within databases are *“predictable and repetitive”* and that performing these tasks on a large scale can be insufficient working. This in turn wastes resources as that effort could be better used in creating more complex and meaningful additions to the database. Craig Mullins(2021) additionally stresses the importance of this as data management within organisations is constantly expanding, this implies automation is of great importance in order to keep systems well maintained. Automation can also reduce cost and time lost through errors. This is because you may only have to make once change to the data to automate it. Rather than multiple manual changes which have a much higher chance of errors occurring.

Throughout this assignment we will be using the following automation techniques, Sequences, … a breakdown of each automation method is included below.

|  |  |
| --- | --- |
| **Automation Method** | **Planned Action** |
| **Sequences** |  |
| seq\_team\_id | Every time a new record is added to the teams table the ID is automatically inserted with the integer of the last plus one, starting from one. |
| seq\_guide\_id | Every time a new record is added to the guides table the ID is automatically inserted with the integer of the last plus one, starting from one hundred. |
| seq\_supervisor\_id | Every time a new record is added to the guides table the ID is automatically inserted with the integer of the last plus one, starting from two hundred. |
| seq\_practice\_id | Every time a new record is added to the practices table the ID is automatically inserted with the integer of the last plus one, starting from three hundred. |
| seq\_ambassador\_id | Every time a new record is added to the ambassador table the ID is automatically inserted with the integer of the last plus one, starting from four hundred. |
| seq\_programme\_id | Every time a new record is added to the programme table the ID is automatically inserted with the integer of the last plus one, starting from five hundred. |
| **Functions** |  |
|  | Subtract employee start date from system date to get time employed variable. |
|  |  |
| **Loops and Ifs** |  |
|  | Loops through all the holiday packages available, has a counter for all in a selected week, a bit like a booking would |
|  | Loops through all the teams to find sizes, if size is smaller than ten then display to the user that more staff needs to be hired in that team. |
| **Procedures** |  |
| **Cursors** |  |

Section 2

Test Plans

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test Title: | | | | |
|  | | | | |
| Test Scope: | | | | |
|  | | | | |
| Testing Approach: | | | | |
|  | | | | |
| Tester: | | | | |
|  | | | | |
| Features Tested: | | | | |
|  | | | | |
| Testing Risks: | | | | |
|  |  | | | |
| Testing Tasks | | | | |
| Test ID Number | Test Description | Expected Results | Actual Results (Tick if matching Expected) | Notes |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

Test Schedule

|  |  |
| --- | --- |
| Test Title | Intended Test Schedule |

Appendix

Appendix 1 – Create Objects

Appendix 2 – Create Tables

Appendix 3 – Table Constraints and Sequences

Appendix 4 – Data Inserts

Appendix 5 – Queries

Appendix 6(to wherever) – Procedures

Appendix ? – All Drops

References

<https://www.talend.com/resources/data-extraction-defined/> [accessed on second commit]

<https://support.microsoft.com/en-us/office/create-a-simple-select-query-de8b1c8d-14e9-4b25-8e22-70888d54de59> [accessed on second commit] <https://whatis.techtarget.com/definition/database-automation> [accessed 01/01/2022]

https://www.bmc.com/blogs/database-automation/#[accessed 01/01/2022]

https://www.dbta.com/Columns/DBA-Corner/Why-DBAs-Must-Automate-to-Survive-149365.aspx[accessed 01/01/2022]

<https://www.red-gate.com/simple-talk/blogs/sql-naming-conventions/>

<https://oracle-base.com/articles/misc/naming-conventions>

<https://docs.oracle.com/cd/E18727_01/doc.121/e12897/T302934T458266.htm>

<https://www.sqlshack.com/learn-sql-naming-conventions/> [accessed 04/01/2022]